Государственное бюджетное образовательное учреждение  
высшего образования Московской области  
«Университет «Дубна»  
Филиал «Котельники»

Кафедра «Информационные технологии в управлении»

КУРСОВАЯ РАБОТА

По дисциплине: Программные технологии интернет

Тема: Разработка веб-приложения «Школьные классы»

Ф.И.О. студента Жукова Анастасия Владиславовна

Группа   ИВТ-31 Направление подготовки «Информатика и вычислительная техника»

Руководитель: к.ф.-м.н., доцент, Орлов Н.Н.    /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

(ученая степень, должность, ФИО) (подпись)

Выполнил: студент(ка)

          ИВТ-31             группы

                3                     курса

Жукова А.В.

(Фамилия И.О. студента)

Дата защиты:

Оценка:

Подпись

(подпись руководителя)

г. Котельники, 2022

Государственное бюджетное образовательное учреждение

высшего образования Московской области

«Университет «Дубна»

Филиал «Котельники»

Кафедра «Информационные технологии в управлении»

УТВЕРЖДАЮ

Заведующий кафедрой

                                      / Орлов Н.Н.   /

(подпись) (ФИО)

ЗАДАНИЕ

на курсовую работу (проект)

Тема: Разработка веб-приложения «Школьные классы»

По дисциплине: Программные технологии интернет

Ф.И.О. студента Жукова Анастасия Владиславовна

Группа   ИВТ-31   Направление подготовки «Информатика и вычислительная техника»

Профильная направленность образовательной программы «Технология разработки

программного обеспечения»

Дата выдачи задания

Дата завершения курсовой работы

Задание выдал \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Орлов Н.Н. /

подпись ФИО

Задание принял \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Жукова А.В. /

подпись ФИО

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc1)

[1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 6](#_Toc2)

[1.1 Язык разметки страницы HTML 6](#_Toc3)

[1.2 Каскадная таблица стилей CSS 6](#_Toc4)

[1.3 Язык программирования JavaScript 7](#_Toc5)

[1.4 Язык программирования PHP 8](#_Toc6)

[1.5 Технология AJAX 9](#_Toc7)

[2 ОСНОВНАЯ ЧАСТЬ 11](#_Toc8)

[2.1 Условия задания 11](#_Toc9)

[2.2 Создание БД на языке SQL 13](#_Toc10)

[2.3 Главная страница веб-приложения 19](#_Toc11)

[2.4 Создание формы для таблицы 25](#_Toc12)

[2.5 Проверка данных на корректность на стороне пользователя 27](#_Toc13)

[2.6 Работа с данными формы на сервере 33](#_Toc14)

[2.7 Создание сводной таблицы 44](#_Toc15)

[ЗАКЛЮЧЕНИЕ 46](#_Toc16)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 48](#_Toc17)

[ПРИЛОЖЕНИЕ А 52](#_Toc18)

[ПРИЛОЖЕНИЕ Б 56](#_Toc19)

[ПРИЛОЖЕНИЕ В 63](#_Toc20)

# **ВВЕДЕНИЕ**

В настоящее время трудно представить жизнь простого человека без интернета. У большого числа людей профессиональная сфера жизненной деятельности связана с работой в этой мировой сети. Из интернета пользователи могут узнать вообще всё что угодно – от погоды на ближайшие дни и общения в социальных сетях до оформления заказа на иностранную продукции из-за границы. Современному человеку, подстроившемуся под эту информационную волну, будет тяжело представить, что около 20-ти лет назад у людей не было такой потребности [1].

Кроме поиска познавательной и научной информации, люди, ко всему прочему, могут использовать интернет, чтобы поиграть в какие-то игры. Сейчас можно играть в браузере в игры, вышедшие около 30-ти лет назад, в большинстве случаев они эмулируются. Или же можно играть в игры по интернету с другими людьми (многопользовательские игры) [2].

Однако есть и люди, использующие интернет не только для проведения там своего свободного времени. Очень много людей ищут и находят в сети работу, те же платформы, на которых есть множество вакансий. Также люди открывают в интернете свое дело, например, продажа каких-то вещей. Способов заработка в интернете предостаточно, и перечислять их будет слишком долго [3].

Еще интернет используют как место, где можно продемонстрировать свои творческие способности и опять же заработать на этом. К примеру, это могут быть какие-то статьи, рисунки, блоги, комиксы, видео, мультфильмы и прочее.

Целью данной курсовой работы является выполнение различных задач, включающих в себя теоретические знания по языку программирования JavaScript, PHP, языку разметки страницы HTML и созданию каскадной таблицы стилей CSS.

Главными задачами курсовой работы являются:

* изучение основ HTML и CSS;
* изучение основ языка программирования JavaScript;
* изучение основ языка программирования PHP;
* работа с базой данных MySQL;
* работа с технологией AJAX.

В процессе написания курсовой работы должны быть освоены следующие компетенции:

* ПК-4 Проектирование и дизайн ИС;
* ПК-5 Организационное и технологическое обеспечение кодирования на языках программирования и модульного тестирования ИС.

В результате выполнения курсовой работы должно получиться небольшое web-приложение, которое будет комбинировать все вышеуказанные задачи.

# **1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

## **1.1 Язык разметки страницы HTML**

HTML (Hypertext Markup Language) не относится к языкам программирования, это язык разметки страницы, который был создан для того, чтобы показывать браузеру как необходимо отображать посещаемые пользователем веб-страницы. В зависимости от предпочтений разработчика, эта разметка может быть очень примитивной или сложной [4].

HTML-документ представляет из себя обычный текстовый документ, который можно создать через простой текстовый редактор, например, Блокнот, или через специальные текстовые приложения, которые будут подсвечивать код в случае каких-то неполадок (Visual Studio Code, Notepad++). Для того, чтобы обозначить, что созданный документ является HTML документом, при наименовании в конце ставят .html.

HTML-документ состоит из древовидной структуры HTML-элементов и текста. У каждого элемента, за редким исключением, есть свой открывающийся и закрывающийся тэг. Открывающийся тэг показывает, откуда начинается элемент, а закрывающий, соответственно, показывает, где он заканчивается. Между двумя этими тэгами находится контент – содержимое элемента [5].

Элементы можно вкладывать друг в друга. Однако важно помнить правило так называемой «матрешки», что если один тэг вкладывается в другой, то на выходе сначала закрывают внутренний тэг, и только после – внешний. Также у элементов можно прописывать атрибуты в виде имя-значение. При помощи атрибутов можно изменять свойства элемента, в котором они прописаны.

## **1.2 Каскадная таблица стилей CSS**

CSS (Cascading Style Sheets) представляет собой язык таблиц стилей, с помощью которого можно закрепить созданный стиль (к примеру, цвет фона или шрифт текста) на конкретном структурированном документе HTML и не только. Чаще всего CSS используют для создания и изменения стиля элементов на веб-странице или же пользовательских интерфейсов, созданных при помощи разметки HTML [6].

CSS описывает то, как необходимо форматировать элемент при помощи свойств и значений, доступных для конкретного свойства. Объявление стиля состоит из двух частей: селектор и объявление. Объявление состоит из двух частей: имя свойства (например, border-color) и значение свойства (например, red). Селектор говорит браузеру, какой именно элемент форматировать, а в блоке объявления (это код в фигурных скобках) перечисляются форматирующие команды – свойства и их значения [6].

## **1.3 Язык программирования JavaScript**

JavaScript (JS) представляет собой язык программирования, который используют для взаимодействия с HTML документом и который может создать некую интерактивность на веб-странице. Весь код, что пишут в скрипте HTML, заключают в открывающийся и закрывающийся тэги script. JS можно назвать универсальным языком и новички при должном опыте в других языках быстро разберутся и здесь [7-8].

JS сам по себе довольно компактный, но очень гибкий. Разработчики создали большое количество инструментов для облегчения использования языка. К основным преимуществам при использовании JS можно отнести следующее [9]:

* широкая распространенность, то есть любой браузер поддерживает этот язык;
* JS-приложение не требует установки на компьютер пользователя, так как все скрипты выполняются непосредственно в браузере при загрузке главной страницы;
* простота в изучении, а именно – этот язык изучают в школах;
* относится к языкам высокого уровня – это значит, что есть заранее заготовленный набор команд, при помощи которых пишется программный код;
* быстрота для пользователя – это преимущество обусловлено тем, что код выполняется на стороне браузера.

К недостаткам использования языка JS можно отнести следующее [9]:

* низкая безопасность – в сети можно найти много исходного кода уже готовых веб-приложений, ведь некоторые из них находятся в свободном доступе, из-за этого тяжело гарантировать что такое веб-приложение не взломают;
* повсеместное использование – если будет разработано что-то удобнее JS, что устранит большинство недостатков, то надобность в коде, разработанном на JS пропадет и придется переписывать все на новом языке.

## **1.4 Язык программирования PHP**

PHP (Hypertext PreProcessor, «препроцессор гипертекста») – является скриптовым языком программирования с открытым исходным кодом. По изначальной задумке создавался для разработки веб-приложений, но в процессе обновлений стал языком общего назначения [10].

Под основной областью применения этого языка понимают разработку скриптов, которые обрабатываются на стороне сервера, именно поэтому язык считается серверным. PHP используют в разномасштабных проектах, начиная от социальных сетей и игр в интернете, заканчивая довольно крупными веб-приложениями (например, YouTube) [11].

К ключевым особенностям в PHP можно отнести динамическую типизацию, то есть переменные можно не объявлять заранее, что строго требуется делать в языках Си, C++. Из-за этого также одна и та же переменная может быть, как числом, так и строкой. Однако в этом есть и свои недостатки, связанные с тем, что расход оперативной памяти увеличивается, и уменьшается скорость выполнения программного кода. Еще одной ключевой особенностью является то, что PHP – интерпретируемый язык программирования (как и язык программирования Python). Это означает, что код, который пишет программист, при запуске не компилируется (переводится в сущность, понятную машине), а выполняется последовательно, строка за строкой, это достигается благодаря интерпретатору [12].

Если CSS и НТМL-файлы сервер передает сразу клиенту, то PHP-файлы интерпретируются на сервер. Клиент в результате получает не сам код PHP, а результат его выполнения, который, чаще всего, является чистым HTML. Пользователь не сможет увидеть исходной код, и это хорошо с точки зрения безопасности, однако, из-за этого тяжело понять логику, по которой написан сайт [12].

## **1.5 Технология AJAX**

AJAX (Asynchronous JavaScript and XML) – это «асинхронный JavaScript и XML». Технология нужна для работы с сайтом без перезагрузки всей страницы, это достигается благодаря вставке частей кода через запросы на сервер и получения обратного ответа, то есть страница обновляется только частично [13].

Технология AJAX, как указывает первая буква A в ее названии – асинхронна (вручную можно поставить синхронное выполнение), то есть браузер, отослав запрос, может делать что угодно, например, показать сообщение об ожидании ответа, прокручивать страницу. Основные преимущества при использовании AJAX включают [14]:

* снижение трафика (из-за уменьшения объёма передаваемых данных между клиентом и сервером);
* уменьшение нагрузки на сервер (не нужно генерировать всю страницу, а только ту часть, которую нужно обновить);
* увеличение быстродействия и отзывчивости (нет необходимости в полной перезагрузке страницы, достаточно обновить содержимое только отдельных блоков);
* повышение интерактивности (с помощью AJAX можно сразу отображать результаты и сделать ресурс более удобным для пользования).

Весь процесс создания запроса для получения необходимых данных с сервера можно разделить на несколько этапов [15]:

* создание объекта XMLHttpRequest, встроенного во все современные браузеры, с помощью которого делаются HTTP-запросы к серверу без перезагрузки страницы;
* инициализация запроса через метод open;
* проверка этапов состояния выполнения запроса через метод onreadystatechange;
* отправка запроса методом send.

# **2 ОСНОВНАЯ ЧАСТЬ**

## **2.1 Условия задания**

Создать веб-приложение «Школьные классы». Для этого необходимо выполнить следующие действия:

1. Создать в СУБД MySQL базу данных «Школьные классы», состоящую из 4–х связанных таблиц, представленных ниже:

**Работники школы:**

|  |  |  |  |
| --- | --- | --- | --- |
| Табельный № | ФИО учителя | Должность /профессия | Пол |
| 100[[1]](#footnote-2) | Петров Станислав Васильевич | Учитель | М |
| 101 | Петрова Валентина Григорьевна | Учитель | Ж |
| 102 | Рыбакова Анна Ивановна | Завуч | Ж |
| 103 | Федоров Юрий Васильевич | Директор | М |
| 104 | Смирнов Антон Юрьевич | Учитель | М |

**Классы:**

|  |  |  |  |
| --- | --- | --- | --- |
| № класса | Табельный № учителя | Специализация класса | Классная комната |
| 10А | 101 | Иностранные языки (английский и немецкий) | 203 |
| 11Б | 100 | Математика и физика | 212 |

**Школьники:**

|  |  |  |  |
| --- | --- | --- | --- |
| № класса | Порядковый № в журнале | ФИО школьника | Пол |
| 10А | 1 | Иванов Сергей Петрович | М |
| 10А | 2 | Костин Петр Васильевич | М |
| 10А | 3 | Матросова Елена Ивановна | Ж |

**Школьники1:**

|  |  |  |  |
| --- | --- | --- | --- |
| № класса | Порядковый № в журнале | ФИО школьника | Пол |
| 11Б | 1 | Богданов Юрий Сергеевич | М |
| 11Б | 2 | Потапова Юлия Петровна | Ж |
| 11Б | 3 | Сорокина Ольга Петровна | Ж |
| 11Б | 4 | Сидоров Андрей Петрович | М |

1. Создать на веб-странице некоторую область, в которой будут содержаться ссылки для получения данных о таблицах с базы данных MySQL.
2. Обеспечить работу базы данных с веб-сервером посредством языка PHP.
3. При нажатии на одну из ссылок должна выводиться таблица на той же веб-странице, где находятся ссылки.
4. Для каждой таблицы необходимо создать форму для работы с данными.
5. Написать JavaScript код, проверяющий соответствие введенных в поля данных требуемым ограничениями.
6. Написать операции добавления, обновления и удаления данных на языке PHP, которые заполняются пользователем в заранее созданной форме и осуществить проверку на возможность выполнения этих операций.
7. Реализовать сводную таблицу на основе созданных ранее таблиц на языке SQL для получения отчета.
8. Написать PHP код, который проверяет полученные данные на сервере по заранее созданному шаблону.
9. Осуществить технологию динамического обращения к серверу «на лету», без перезагрузки всей страницы полностью (далее, AJAX).

## **2.2 Создание БД на языке SQL**

Функционирование веб-приложения с интернетом будет произведено с помощью бесплатного веб-сервера Apache, который был включен в бесплатный пакет XAMPP. В самом начале разработки приложения необходимо создать БД «Школьные классы» с четырьмя таблицами из условия. Создание будет произведено с помощью языка запросов SQL и процедурный стиль взаимодействия PHP с сервером MySQL. Ниже приведен листинг программного кода на выполнение данной задачи (весь код находится в скрипте create.php) [16-17].

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

</head>

<body>

<?php

include\_once "login.php";

$conn = new mysqli($server, $user, $password, $dbname);

if($conn->connect\_error) die("Во время подключения к веб-серверу произошла ошибка:" . $conn->connect\_error . "<br>");

else {

echo("Успешное подключение к серверу произведено!" . "<br>");

$query="DROP TABLE IF EXISTS `Работники школы`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно удалить таблицу \"Работники школы\":<br>\n".$conn->error."</p>\n";

else {

$query="CREATE TABLE IF NOT EXISTS `Работники школы`

(`Табельный номер` INT(3) PRIMARY KEY,

`ФИО учителя` VARCHAR(40) NOT NULL,

`Должность/профессия` VARCHAR(15) NOT NULL,

`Пол` VARCHAR(1) NOT NULL);";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно создать таблицу \"Работники школы\":<br>\n".$conn->error."</p>\n";

else

echo "<p>Таблица \"Работники школы\" создана</p>\n";

}

$insert\_Table = "INSERT IGNORE INTO `Работники школы` (`Табельный номер`, `ФИО учителя`, `Должность/профессия`, `Пол`) VALUES

(100, 'Петров Станислав Васильевич', 'Учитель', 'М'),

(101, 'Петрова Валентина Григорьевна', 'Учитель', 'Ж'),

(102, 'Рыбакова Анна Ивановна', 'Завуч', 'Ж'),

(103, 'Федеров Юрий Васильевич', 'Директор', 'М'),

(104, 'Смирнов Антон Юрьевич', 'Учитель', 'М');";

$result=$conn->query($insert\_Table);

if(!$result) die("Во время добавления записей в таблицу \"Работники школы\" произошла ошибка:" . $conn->error . "<br>");

else echo("В таблицу \"Работники школы\" успешно добавлены записи в кол-ве " . $conn->affected\_rows . " штук(и)" . "<br>");

$query="DROP TABLE IF EXISTS `Классы`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно удалить таблицу \"Классы\":<br>\n".$conn->error."</p>\n";

else {

$query="CREATE TABLE IF NOT EXISTS `Классы`

(`№ класса` VARCHAR(3) NOT NULL,

`Табельный № учителя` INT(3) NOT NULL,

`Специализация класса` VARCHAR(40) NOT NULL,

`Классная комната` INT(3) NOT NULL,

PRIMARY KEY (`№ класса`, `Табельный № учителя`),

FOREIGN KEY (`Табельный № учителя`) REFERENCES `Работники школы` (`Табельный номер`) ON DELETE RESTRICT ON UPDATE CASCADE);";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно создать таблицу \"Классы\":<br>\n".$conn->error."</p>\n";

else

echo "<p>Таблица \"Классы\" создана</p>\n";

}

$insert\_Table = "INSERT IGNORE INTO `Классы` (`№ класса`, `Табельный № учителя`, `Специализация класса`, `Классная комната`) VALUES

('10А', 101, 'Иностранные языки (английский и немецкий', 203),

('11Б', 100, 'Математика и физика', 212);";

$result=$conn->query($insert\_Table);

if(!$result) die("Во время добавления записей в таблицу \"Классы\" произошла ошибка:" . $conn->error . "<br>");

else echo("В таблицу \"Классы\" успешно добавлены записи в кол-ве " . $conn->affected\_rows . " штук(и)" . "<br>");

$query="DROP TABLE IF EXISTS `Школьники`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно удалить таблицу \"Школьники\":<br>\n".$conn->error."</p>\n";

else {

$query="CREATE TABLE IF NOT EXISTS `Школьники`

(`№ класса` VARCHAR(3) NOT NULL,

`Порядковый № в журнале` INT(2) NOT NULL,

`ФИО школьника` VARCHAR(40) NOT NULL,

`Пол` VARCHAR(1) NOT NULL,

PRIMARY KEY (`№ класса`, `Порядковый № в журнале`),

FOREIGN KEY (`№ класса`) REFERENCES `Классы` (`№ класса`) ON DELETE RESTRICT ON UPDATE CASCADE);";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно создать таблицу \"Школьники\":<br>\n".$conn->error."</p>\n";

else

echo "<p>Таблица \"Школьники\" создана</p>\n";

}

$insert\_Table = "INSERT IGNORE INTO `Школьники` (`№ класса`, `Порядковый № в журнале`, `ФИО школьника`, `Пол`) VALUES

('10А', 1, 'Иванов Сергей Петрович', 'М'),

('10А', 2, 'Костин Петр Васильевич', 'М'),

('10А', 3, 'Матросова Елена Ивановна', 'Ж');";

$result=$conn->query($insert\_Table);

if(!$result) die("Во время добавления записей в таблицу \"Школьники\" произошла ошибка:" . $conn->error . "<br>");

else echo("В таблицу \"Школьники\" успешно добавлены записи в кол-ве " . $conn->affected\_rows . " штук(и)" . "<br>");

$query="DROP TABLE IF EXISTS `Школьники1`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно удалить таблицу \"Школьники1\":<br>\n".$conn->error."</p>\n";

else {

$query="CREATE TABLE IF NOT EXISTS `Школьники1`

(`№ класса` VARCHAR(3) NOT NULL,

`Порядковый № в журнале` INT(2) NOT NULL,

`ФИО школьника` VARCHAR(40) NOT NULL,

`Пол` VARCHAR(1) NOT NULL,

PRIMARY KEY (`№ класса`, `Порядковый № в журнале`),

FOREIGN KEY (`№ класса`) REFERENCES `Классы` (`№ класса`) ON DELETE RESTRICT ON UPDATE CASCADE);";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно создать таблицу \"Школьники1\":<br>\n".$conn->error."</p>\n";

else

echo "<p>Таблица \"Школьники1\" создана</p>\n";

}

$insert\_Table = "INSERT IGNORE INTO `Школьники1` (`№ класса`, `Порядковый № в журнале`, `ФИО школьника`, `Пол`) VALUES

('11Б', 1, 'Богданов Юрий Сергеевич', 'М'),

('11Б', 2, 'Потапова Юлия Петровна', 'Ж'),

('11Б', 3, 'Сорокина Ольга Петровна', 'Ж'),

('11Б', 4, 'Сидоров Андрей петрович', 'М');";

$result=$conn->query($insert\_Table);

if(!$result) die("Во время добавления записей в таблицу \"Школьники1\" произошла ошибка:" . $conn->error . "<br>");

else echo("В таблицу \"Школьники1\" успешно добавлены записи в кол-ве " . $conn->affected\_rows . " штук(и)" . "<br>");

$conn->close();

}

?>

</body>

</html>

В самом начале происходит подключение к серверу MySQL (параметры подключения находятся в файле login.php), а в самом конце отключение с использованием функции «mysqli\_close», аргументом в которой является адрес подключения. При запуске данного файла через веб-сервер будет показан результат, соответствующий рисунку 2.2.1 [18].

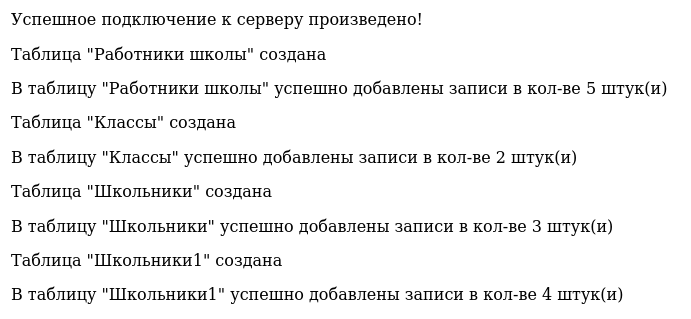


Рисунок 2.2.1 – Создание БД

Для того, чтобы удостовериться в том, что БД с таблицами была создана, необходимо зайти в веб-приложение PhpMyAdmin и проверить наличие созданной базы в перечне имеющихся. Результат из программы представлен на рисунке 2.2.2 [19].

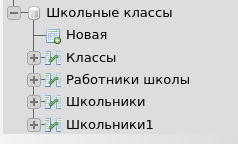


Рисунок 2.2.2 – Проверка успешного создания БД

При нажатии на любую из трех созданных таблиц в данной БД пользователь увидит столбцы, созданные через конструкцию CREATE TABLE языка SQL, и строки с данными, занесенными в таблицу через конструкцию INSERT INTO языка SQL. Следующие рисунки 2.2.3, 2.2.4, 2.2.5 и 2.2.6 это демонстрируют.

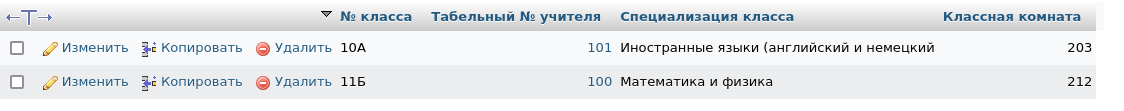


Рисунок 2.2.3 – Проверка таблицы «Классы»



Рисунок 2.2.4 – Проверка таблицы «Работники школы»

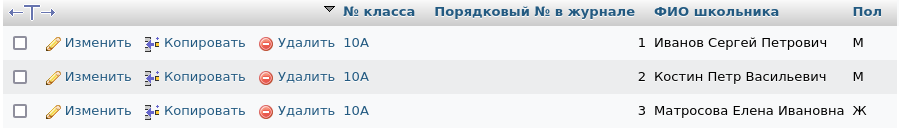


Рисунок 2.2.5 – Проверка таблицы «Школьники»



Рисунок 2.2.6 – Проверка таблицы «Школьники1»

## **2.3 Главная страница веб-приложения**

По условию задания на главной странице должна быть область, в которой будут находиться ссылки (кнопки), при нажатии на которые в другую область будет перемещаться таблица при помощи технологии AJAX, также синхронно с появлением таблицы в третьей области должна появиться форма при помощи технологии AJAX, содержащая поля, являющиеся наименованием ячеек из выбранной таблицы. Также есть еще две области, в первой находится название создаваемого приложения, а во второй – ФИО создателя. Главная страница была создана при помощи языка разметки страницы HTML, каскадной таблицы стилей CSS. Запросы на веб-сервер по технологии AJAX для получения соответствующей таблицы и формы были сделаны при помощи языка программирования JS. Ниже представлен листинг программного кода на создание главной страницы (весь код данной задачи расположен в скрипте index.php).

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Лицей имени Жуковой</title>

<link rel="stylesheet" href="default.css">

<link rel="stylesheet" href="style.css">

<script>

var XMLHttpRequestObject = false;

if (window.XMLHttpRequest)

XMLHttpRequestObject = new XMLHttpRequest();

else if (window.ActiveXObject) {

XMLHttpRequestObject = new ActiveXObject("Microsoft.XMLHTTP");

}

function postDate(file, data, callback){

if(XMLHttpRequestObject){

XMLHttpRequestObject.open("POST", file, true);

XMLHttpRequestObject.onreadystatechange = function(){

if(XMLHttpRequestObject.status == 200 && XMLHttpRequestObject.readyState == 4 && callback){

callback(XMLHttpRequestObject.responseText);

}

}

XMLHttpRequestObject.send(data);

}

else document.write("Что-то пошло не так ...");

}

function getDate(file, id) {

if (XMLHttpRequestObject) {

var obj = document.getElementById(id);

XMLHttpRequestObject.open("GET", file, false);

XMLHttpRequestObject.onreadystatechange = function () {

if (XMLHttpRequestObject.readyState == 4 && XMLHttpRequestObject.status == 200) {

obj.innerHTML = XMLHttpRequestObject.responseText;

}

}

XMLHttpRequestObject.send(null);

}};

function getStr(){

var selct = document.getElementById("tbl")

selct.addEventListener('click', (e) => {

let cell = e.target;

let t = cell.parentElement;

let r = t.parentElement.children;

for (rowCount = 1; rowCount < r.length; rowCount++) {

if (r[rowCount] === t) {

let d = r[rowCount].getElementsByTagName("td");

document.getElementById("one").value = d[0].innerHTML;

document.getElementById("one\_Old").value = d[0].innerHTML;

document.getElementById("two").value = d[1].innerHTML;

document.getElementById("two\_Old").value = d[1].innerHTML;

document.getElementById("tree").value = d[2].innerHTML;

document.getElementById("tree\_Old").value = d[2].innerHTML;

document.getElementById("four").value = d[3].innerHTML;

document.getElementById("four\_Old").value = d[3].innerHTML;

return;

}

}

document.getElementById("tblName").value=document.querySelector('caption').innerHTML;

rowCount = -1

})

};

function f(data){

document.getElementsByTagName("article")[0].innerHTML = data;

}

function f1(op) {

tblName.setAttribute('value', document.querySelector('caption').innerHTML);

tbl = document.querySelector('caption').innerHTML;

if(op=="update"){

one = document.getElementById("one\_Old").value;

two = document.getElementById("two\_Old").value;

tree = document.getElementById("tree\_Old").value;

four = document.getElementById("four\_Old").value;

}

else{

one = document.getElementById("one").value;

two = document.getElementById("two").value;

tree = document.getElementById("tree").value;

four = document.getElementById("four").value;

}

//var t = document.querySelector('caption').innerHTML;

if (tbl == 'Классы') {

var errorText = "";

data = one;

var reg = new RegExp("[a-zA-Z](...)");

if(reg.test(data) == true || data.length < 2 || data.length > 3){

errorText += "Поле \"№ класса\" должно быть строкой из русских букв и цифр до 3-х символов\n";

}

data = Number.parseInt(two);

if(isNaN(data) == true || data.length < 100 || data.length > 150){

errorText += "Поле \"Табельный № учителя\" должно быть числом в диапазоне от 100 до 150\n";

}

data = tree;

if(reg.test(data) == true || data.length < 1 || data.length > 40){

errorText += "Поле \"Специализация класса\" должно быть строкой из русских букв до 40-ка символов\n";

}

data = Number.parseInt(four);

if(isNaN(data) == true || data < 1 || data > 999){

errorText += "Поле \"Классная комната\" должно быть числом в диапазоне от 1 до 999\n";

}

if(errorText != "") {

alert(errorText);

}

else{

Ok.setAttribute('value', op);

}

}

else if (tbl == 'Работники школы'){

var errorText = "";

data = Number.parseInt(one);

var reg = new RegExp("[a-zA-Z](...)");

if(isNaN(data) == true || data < 1 || data > 150){

errorText += "Поле \"Табельный номер\" должно быть числом в диапазоне от 100 до 150" + "<br></p>";

}

data = two;

if(reg.test(data) == true || data.length < 1 || data.length > 40){

errorText += "Поле \"ФИО учителя\" должно быть строкой из русских букв до 40-ка символов\n";

}

data = tree;

if(reg.test(data) == true || data.length < 1 || data.length > 15){

errorText += "Поле \"Должность/профессия\" должно быть строкой из русских букв до 15-ти символов\n";

}

data = four;

reg = new RegExp("[a-zA-Zа-я](...)");

if(reg.test(data) == true || data.length < 1 || data.length > 1){

errorText += "Поле \"Пол\" должно быть строкой из русских букв из 1 символа\n";

}

if(errorText != "") {

alert(errorText);

}

else{

Ok.setAttribute('value', op);

}

}

else if (tbl == 'Школьники'||tbl == 'Школьники1'){

var errorText = "";

data = one;

var reg = new RegExp("[a-zA-Z](...)");

if(reg.test(data) == true || (data != '10А' && tbl == 'Школьники')){

errorText += "Поле \"№ класса\" в таблице 'Школьники' должно иметь значение '10A'\n";

}

else if(reg.test(data) == true || (data != '11Б' && tbl == 'Школьники1')){

errorText += "Поле \"№ класса\" в таблице 'Школьники1' должно иметь значение '11Б'\n";

}

data = Number.parseInt(two);

if(isNaN(data) == true || data < 1 || data > 50){

errorText += "Поле \"Порядковый № в журнале\" должно быть числом в диапазоне от 1 до 50\n";

}

data = tree;

if(reg.test(data) == true || data.length < 1 || data.length > 40){

errorText += "Поле \"ФИО школьника\" должно быть строкой из русских букв до 40-ка символов\n";

}

data = four;

reg = new RegExp("[a-zA-Zа-я](...)");

if(reg.test(data) == true || data.length < 1 || data.length > 1){

errorText += "Поле \"Пол\" должно быть строкой из русских букв из 1 символа\n";

}

if(errorText != "") alert(errorText);

else Ok.setAttribute('value', op);

}

};

</script>

</head>

<body>

<header>

<img class="logo" src="https://cdn-icons-png.flaticon.com/512/4645/4645232.png" alt="logo">

<h2>Лицей имени Жуковой</h2>

<nav>

<ul class="nav">

<li><input type="submit" class="nav\_\_item" value="Работники школы" onclick="getDate('selectRB.php','tables');getDate('./forms/formRB.txt','form');getStr();"></li>

<li><input type="submit" class="nav\_\_item" value="Классы" onclick="getDate('selectKL.php','tables');getDate('./forms/formKL.txt','form');getStr();"></li>

<li><input type="submit" class="nav\_\_item" value="Школьники" onclick="getDate('selectSHK.php','tables');getDate('./forms/formSHK.txt','form');getStr();"></li>

<li><input type="submit" class="nav\_\_item" value="Школьники1" onclick="getDate('selectSHK1.php','tables');getDate('./forms/formSHK.txt','form');getStr();"></li>

</ul>

</nav>

<div class="info">

<ul>

<li class="info\_\_list">

</li>

<li class="info\_list">Телефон: 88005553535</li>

<li class="info\_list">Почта: JuckLic@mail.ru</li>

<a class="adress" href="https://goo.gl/maps/ibCerVgjkaHe86rn8">

Котельники,

Московская обл.,

140053

<img class="adress\_\_icon" src="https://cdn-icons-png.flaticon.com/512/684/684908.png" alt="">

</a>

</ul>

</div>

</header>

<aside id="form">

</aside>

<main>

<section class="vid" id="vid">

<input type="submit" class="nav\_\_item" value="Заполнить БД стандартно" onclick="getDate('create.php','vid')">

<?php include\_once 'IDU.php';?>

</section>

<section class="tables">

<article id="tables">

</article>

</section>

</main>

<footer>

<small>© 2022 "Лицей и.Жуковой". Жукова Анастасия Владиславовна</small>

</footer>

</body>

</html>

Результат исполнения кода из этого скрипта изображен на рисунке 2.3.1. Дизайн для приложения является очень простым и был выбран на усмотрение создателя. Программный код CSS находится в приложении А.

Важно отметить работу кнопок отображения таблиц. При нажатии на одну из четырех кнопок, соответствующих ранее созданным таблицам, будут вызваны последовательно несколько функций, которые и осуществят XMLHttpRequest запрос на сервер, возьмут оттуда содержимое необходимых файлов и вернут их пользователю в нужные области. [20].

Еще стоит отметить обработчик события, который нужен для переноса записи из таблицы в поля формы. Если пользователь кликнет куда-то кроме таблицы, в области, куда она будет перенесена при нажатии на кнопку, в консоли браузера выведется сообщение, что необходимо кликать исключительно по таблице для корректного функционирования обработчика. Также было предусмотрено ограничение работы обработчика при вызове «Отчета» – во всех таблицах, используемых в нем, обработчик не будет работать [21].

## **2.4 Создание формы для таблицы**

Для каждой таблицы будет создана своя собственная форма со своими собственными полями, однако, на всех формах будут одинаковые 3 кнопки, соответствующие выбору пользователя: добавить запись в таблицу, удалить запись из таблицы и изменить выбранную запись в таблице. Так как принцип, по которому созданы формы один и тот же, программный код в эту часть курсовой работы будет прописан только для формы к таблице «Классы», остальные формы будут находиться в Приложении В. Ниже представлен листинг кода для создания формы (файлы с формами для таблиц находятся в папке forms).

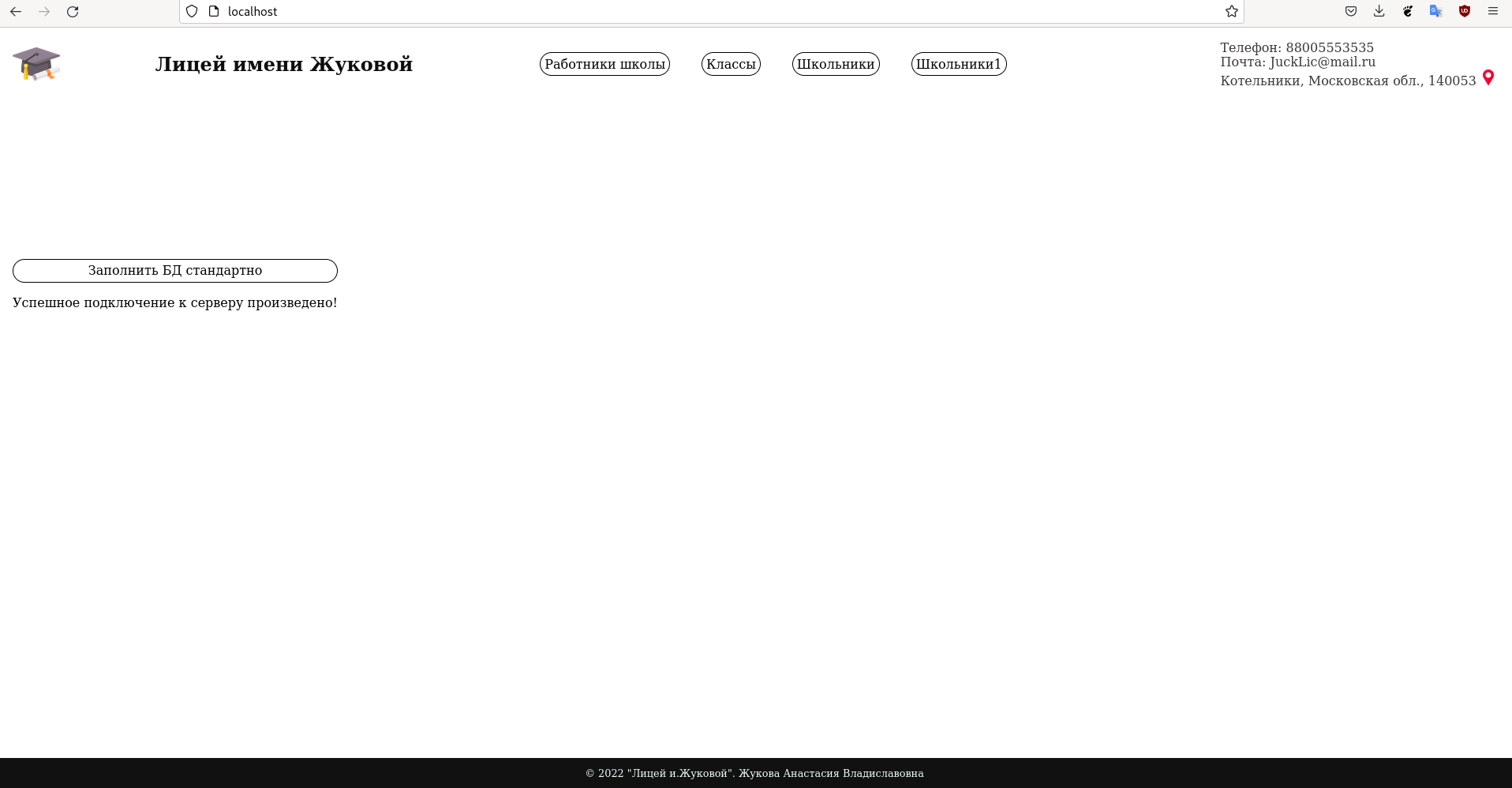


Рисунок 2.3.1 – Внешний вид приложения

<form method="post">

<input type="hidden" id="Ok" name="Ok">

<input type="hidden" id="tblName" name="tblName">

<input type="hidden" id="one\_Old" name="one\_Old">

<input type="hidden" id="two\_Old" name="two\_Old">

<input type="hidden" id="tree\_Old" name="tree\_Old">

<input type="hidden" id="four\_Old" name="four\_Old">

<input name="one" id="one" placeholder="№ класса" required>

<input name="two" id="two" placeholder="Табельный № учителя" required>

<input name="tree" id="tree" placeholder="Специализация класса" required>

<input name="four" id="four" placeholder="Классная комната" required>

<button onclick="f1('insert')">Добавить выбранную запись</button>

<button onclick="f1('update')">Изменить выбранную запись</button>

<button onclick="f1('delete')">Удалить выбранную запись</button>

</form>

Стоит отметить следующее. Первое – это поля типа hidden, которые используются исключительно при «Обновлении записи» и необходимы для того, чтобы запомнить старые значения, которые будет необходимо заменить на новые значения, введенные пользователем в форму. Второе – это кнопки типа «button» для основных операций пользователя, а не «submit». Это очень важно, так как по заданию требуется сделать так, чтобы таблицы обновлялись без перезагрузки страницы, то есть по технологии AJAX, которая будет обновлять только часть веб-страницы. Если бы кнопки были типа «submit», то по окончании выполнения выбранной операции (создание, удаление, обновление) вся страница была бы целиком автоматически перезагружена, чего не должно происходить (если сайт будет делать что-то сложное или тяжелое с точки зрения вычислительной мощности сервера, то при долгом ожидании браузер предупредит о том, что ответа от сервера не поступает в течение долгого времени и предложит закрыть веб-сайт или же сайт вовсе может зависнуть). Форма представлена на рисунке 2.4.1 [22].

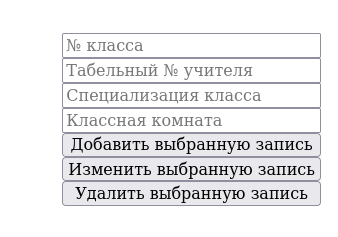


Рисунок 2.4.1 – Форма для таблицы «Классы»

Пользователь может перенести данные в форму из таблицы при помощи нажатия левой кнопки мыши на строку или же ввести данные вручную, но такое действие не сработает, если будет необходимо сделать «Обновление записи», оно строго привязано к обработчику событий.

## **2.5 Проверка данных на корректность на стороне пользователя**

Из программного кода видно, что при нажатии на одну из трех кнопок вызывается функция **«f1(‘’)»**. Ее задача проверить данные, полученные от пользователя через форму, на корректность. Важно, что через JS проверяется корректность данных на стороне клиента, а через PHP точно такая же проверка происходит уже на стороне самого сервера. Все скрипты с проверкой через JS прикрепляются к основному файлу «index.php» через тэг «script» [23].

Ниже представлен листинг, демонстрирующий работу функции.

function f1(op) {

//определяем, какая таблица сейчас открыта и записываем значение в скрытое поле

tblName.setAttribute('value', document.querySelector('caption').innerHTML);

//так же создаем отдельную переменную для названия таблицы, чтобы функция поняла какой шаблон проверки использовать

tbl = document.querySelector('caption').innerHTML;

one = document.getElementById("one").value;

two = document.getElementById("two").value;

tree = document.getElementById("tree").value;

four = document.getElementById("four").value;

if (tbl == 'Классы') {

var errorText = "";

data = one;

var reg = new RegExp("[a-zA-Z](...)");

if(reg.test(data) == true || data.length < 2 || data.length > 3){

errorText += "Поле \"№ класса\" должно быть строкой из русских букв и цифр до 3-х символов\n";

}

data = Number.parseInt(two);

if(isNaN(data) == true || data.length < 100 || data.length > 150){

errorText += "Поле \"Табельный № учителя\" должно быть числом в диапазоне от 100 до 150\n";

}

data = tree;

if(reg.test(data) == true || data.length < 1 || data.length > 40){

errorText += "Поле \"Специализация класса\" должно быть строкой из русских букв до 40-ка символов\n";

}

data = Number.parseInt(four);

if(isNaN(data) == true || data < 1 || data > 999){

errorText += "Поле \"Классная комната\" должно быть числом в диапазоне от 1 до 999\n";

}

if(errorText != "") {

alert(errorText);

}

else{

Ok.setAttribute('value', op);

}

}

else if (tbl == 'Работники школы'){

var errorText = "";

data = Number.parseInt(one);

var reg = new RegExp("[a-zA-Z](...)");

if(isNaN(data) == true || data < 1 || data > 150){

errorText += "Поле \"Табельный номер\" должно быть числом в диапазоне от 100 до 150" + "<br></p>";

}

data = two;

if(reg.test(data) == true || data.length < 1 || data.length > 40){

errorText += "Поле \"ФИО учителя\" должно быть строкой из русских букв до 40-ка символов\n";

}

data = tree;

if(reg.test(data) == true || data.length < 1 || data.length > 15){

errorText += "Поле \"Должность/профессия\" должно быть строкой из русских букв до 15-ти символов\n";

}

data = four;

reg = new RegExp("[a-zA-Zа-я](...)");

if(reg.test(data) == true || data.length < 1 || data.length > 1){

errorText += "Поле \"Пол\" должно быть строкой из русских букв из 1 символа\n";

}

if(errorText != "") {

alert(errorText);

}

else{

Ok.setAttribute('value', op);

}

}

else if (tbl == 'Школьники'||tbl == 'Школьники1'){

var errorText = "";

data = one;

var reg = new RegExp("[a-zA-Z](...)");

if(reg.test(data) == true || (data != '10А' && tbl == 'Школьники')){

errorText += "Поле \"№ класса\" в таблице 'Школьники' должно иметь значение '10A'\n";

}

else if(reg.test(data) == true || (data != '11Б' && tbl == 'Школьники1')){

errorText += "Поле \"№ класса\" в таблице 'Школьники1' должно иметь значение '11Б'\n";

}

data = Number.parseInt(two);

if(isNaN(data) == true || data < 1 || data > 50){

errorText += "Поле \"Порядковый № в журнале\" должно быть числом в диапазоне от 1 до 50\n";

}

data = tree;

if(reg.test(data) == true || data.length < 1 || data.length > 40){

errorText += "Поле \"ФИО школьника\" должно быть строкой из русских букв до 40-ка символов\n";

}

data = four;

reg = new RegExp("[a-zA-Zа-я](...)");

if(reg.test(data) == true || data.length < 1 || data.length > 1){

errorText += "Поле \"Пол\" должно быть строкой из русских букв из 1 символа\n";

}

if(errorText != "") alert(errorText);

else Ok.setAttribute('value', op);

}

};

В аргументы функции передается всего лишь один параметр, сообщающий ей о том, какая операция была выбрана пользователем (в дальнейшем это понадобится).

Так как функция вызывается при нажатии на кнопку, далее в скрытое поле формы ввода записывается название таблицы. Так же создается отдельная переменная, содержащая в себе название таблицы. Данные манипуляции нужны для того, чтобы исключить возможные ошибки связанные с отсутствием таблицы в базе данных, ведь если таблица не была открыта, дальнейшая проверка полей произведена не будет, так как далее код написан исходя из наличия таблицы на странице.

Далее открывается конструкция if-else, которая проверяет название таблицы и выбирает нужный шаблон проверки в соответствии с ней.

Внутри каждого блока конструкции if-else создается переменная, которая будет записывать в себя ошибки, если они возникнут. Условие if проверяет каждое поле на корректность введенного значения.

Для чисел это проверка при помощи функции isNaN на то, что введенное значение – число, а также проверка того, что значение находится в установленном диапазоне [24].

Для строковых значений требуется сделать проверку через регулярные выражения. Для этого создается объект класса «RegEx», которому передается регулярное выражение, по которому будет происходить дальнейшая проверка. В данном коде в качестве регулярного выражения используются все буквы английского алфавита обоих регистров, и поиск будет происходить по всей длине введенной строки, а не до первого соответствия. В условии if для таких полей используется метод test, который ищет соответствие введенного значения в поле формы с тем, что было указано в качестве регулярного выражения, и, если эту функция вернула истину, значит, введено некорректное значение, еще учитывается диапазон, в котором пользователь ввел значение, если оно выходит за заданное значение, значит, введена некорректная строка [25].

Если после проверки полей переменная **errorText** не осталась пустой, то пользователю сообщаются ошибки при помощи передачи переменной **errorText** функции **alert()**. Если же переменная **errorText** осталась пустой, то скрытому полю формы ввода данных **«Ok»** присваивается значение переменной, переданной функции **f1()**. Это действие является «маркером» для php-скрипта **«IDU.php»**, что данные введены корректно и можно отправлять их на сервер, а так же указывает, что именно нужно будет сделать с данными: обновить, удалить или добавить в базу данных[26].

Ниже на рисунке 2.5.1 представлено то, что получит пользователь, если попробует отправить пустую форму или введет не все данные для отправки.

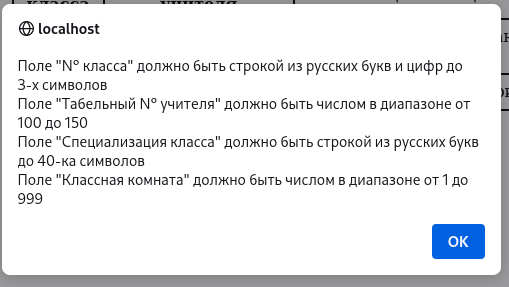


Рисунок 2.5.1 – Пример некорректных данных формы «Квартиры»

## **2.6 Работа с данными формы на сервере**

После того, как данные успешно прошли проверку на корректность со стороны клиента, они передаются на сервер. Здесь они будут проверены еще раз уже через PHP, и в случае успеха будет произведен запрос SQL, который либо создаст новую запись, либо удалит или обновит выбранную. Ниже представлен листинг и объяснение программного кода (программный код расположен в файле **IDU.php**).

<?php

function ValidateFlats($n1, $n2, $n3, $n4,$tbl){

$errorTxt = "";

if($tbl=="Классы"){

if(strlen($n1) < 3 or strlen($n1) > 5 or preg\_match("[a-zA-Z]", $n1) == 1){

$errorTxt .= "<p>Поле \"№ класса\" ".$n1." должно быть вида '11A' и состоять из цифр и букв русского алфавита" . "<br></p>";

}

if(is\_numeric($n2) == false or strlen($n2) < 3 or strlen($n2) > 3 or $n2 < 100 or $n2 > 150){

$errorTxt .= "<p>Поле \"Табельный № учителя\" должно быть числом в диапазоне от 1 до 150" . "<br></p>";

}

//т.к. значение в поле - русские буквы, uft-8 кодирует их по 2 байта

if(strlen($n3) < 2 or strlen($n3) > 80 or preg\_match("[a-zA-Z]", $n3) == 1){

$errorTxt .= "<p>Поле \"Специализация класса\" может состоять только из русских букв"

."и находиться в диапазоне от 1-го до 40-ка символов" . "<br></p>";

}

if(is\_numeric($n4) == false or strlen($n4) < 1 or strlen($n4) > 3 or $n4 < 1 or $n4 > 999){

$errorTxt .= "<p>Поле \"Классная комната\" должно быть числом в диапазоне от 1 до 999" . "<br></p>";

}

}

else if($tbl=="Работники школы"){

if(is\_numeric($n1) == false or strlen($n1) < 3 or strlen($n1) > 3 or $n1 < 100 or $n1 > 150){

$errorTxt .= "<p>Поле \"Табельный номер\" ".$n1." должно быть числом в диапазоне от 1 до 150" . "<br></p>";

}

if(strlen($n2) < 2 or strlen($n2) > 80 or preg\_match("[a-zA-Z]", $n2) == 1){

$errorTxt .= "<p>Поле \"ФИО учителя\" может состоять только из русских букв"

." и находиться в диапазоне от 1-го до 40-ка символов" . "<br></p>";

}

//т.к. значение в поле - русские буквы, uft-8 кодирует их по 2 байта

if(strlen($n3) < 2 or strlen($n3) > 30 or preg\_match("[a-zA-Z]", $n3) == 1){

$errorTxt .= "<p>Поле \"Должность/профессия\" может состоять только из русских букв"

." и находиться в диапазоне от 1-го до 15-ти символов" . "<br></p>";

}

if(strlen($n4) < 2 or strlen($n4) > 2 or preg\_match("[a-zA-Zа-я]", $n4)== 1){

$errorTxt .= "<p>Поле \"Пол\" должно быть символом русского алфавита длинной не больше 1" . "<br></p>";

}

}

else if($tbl=="Школьники"||$tbl=="Школьники1"){

if(strlen($n1) < 3 or strlen($n1) > 5 or preg\_match("[a-zA-Zа-я]", $n1)== 1 or ($tbl=="Школьники" and $n1 !="10А")){

$errorTxt .= "<p>Поле \"№ класса\" ".$n1." таблицы ".$tbl." должно иметь значение '10A' и состоять из цифр и букв русского алфавита" . "<br></p>";

}

else if(strlen($n1) < 3 or strlen($n1) > 5 or preg\_match("[a-zA-Zа-я]", $n1)== 1 or ($tbl=="Школьники1" and $n1 !="11Б")){

$errorTxt .= "<p>Поле \"№ класса\" ".$n1." таблицы ".$tbl." должно иметь значение '11Б' и состоять из цифр и букв русского алфавита" . "<br></p>";

}

//т.к. значение в поле - русские буквы, uft-8 кодирует их по 2 байта

if(is\_numeric($n2) == false or strlen($n2) < 1 or strlen($n2) > 2 or $n2 < 1 or $n2 > 50){

$errorTxt .= "<p>Поле \"Порядковый № в журнале\" должно быть числом в диапазоне от 1 до 50" . "<br></p>";

}

if(strlen($n3) < 2 or strlen($n3) > 80 or preg\_match("[a-zA-Z]", $n3)== 1){

$errorTxt .= "<p>Поле \"ФИО школьника\" должно быть текстом из русского алфавита длинной не больше 40 символов" . "<br></p>";

}

if(strlen($n4) < 2 or strlen($n4) > 2 or preg\_match("[a-zA-Zа-я]", $n4)== 1){

$errorTxt .= "<p>Поле \"Пол\" должно быть символом русского алфавита длинной не больше 1" . "<br></p>";

}

}

return $errorTxt;

}

include\_once "login.php";

$conn = new mysqli($server, $user, $password, $dbname);

if($conn->connect\_error) echo ("<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."<br></p>\n");

else {

echo("<p>Успешное подключение к серверу произведено!" . "<br></p>\n");

$Ok = $\_POST['Ok'];

$tbName = $\_POST['tblName'];

//echo("<p>таблица ".$tbName."</p>");

$one = $\_POST['one'];

$two = $\_POST['two'];

$tree = $\_POST['tree'];

$four = $\_POST['four'];

$err = "";

$err = ValidateFlats($one, $two, $tree,$four,$tbName);

if($err != "") echo($err);

else{

if($Ok=='insert'){

if($tbName == 'Классы'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="INSERT INTO `Классы`(`№ класса`, `Табельный № учителя`, `Специализация класса`, `Классная комната`) ".

"values ('$one', '$two', '$tree', '$four')";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно добавить записи в таблицу \"Классы\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись добавлена в таблицу \"Классы\"</p>\n");

}

else if($tbName == 'Работники школы'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="INSERT INTO `Работники школы`(`Табельный номер`, `ФИО учителя`, `Должность/профессия`, `Пол`) ".

"values ('$one', '$two', '$tree', '$four')";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно добавить записи в таблицу \"Работники школы\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись добавлена в таблицу \"Работники школы\"<br></p>\n");

}

else if($tbName == 'Школьники'|| $tbName == 'Школьники1'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="INSERT INTO `$tbName`(`№ класса`, `Порядковый № в журнале`, `ФИО школьника`, `Пол`) ".

"values ('$one', '$two', '$tree', '$four')";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно добавить записи в таблицу \"".$tbName."\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись добавлена в таблицу \"".$tbName."\"<br></p>\n");

}

}

else if($Ok=='delete'){

if($tbName == 'Классы'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="DELETE FROM `Классы` WHERE `№ класса` = '$one'

AND `Табельный № учителя` = '$two' AND `Специализация класса` = '$tree' AND `Классная комната` = '$four';";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно удалить записи из таблицы \"Классы\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись с ключевым полем ".$one." удалена из таблицы \"Классы\"</p>\n");

}

else if($tbName == 'Работники школы'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="DELETE FROM `Работники школы` WHERE `Табельный номер` = '$one'

AND `ФИО учителя` = '$two' AND `Должность/профессия` = '$tree' AND `Пол` = '$four';";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно удалить записи из таблицы \"Работники школы\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись с ключевым полем ".$one." удалена из таблицы \"Работники школы\"</p>\n");

}

else if($tbName == 'Школьники'|| $tbName == 'Школьники1'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="DELETE FROM `$tbName` WHERE `№ класса` = '$one'

AND `Порядковый № в журнале` = '$two' AND `ФИО школьника` = '$tree' AND `Пол` = '$four';";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно удалить записи из таблицы \"".$tbName."\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись с ключевыми полями ".$one." и ".$two." удалена из таблицы \"".$tbName."\"</p>\n");

}

}

else if($Ok=='update'){

$one\_Old = $\_POST['one\_Old'];

$two\_Old = $\_POST['two\_Old'];

$tree\_Old = $\_POST['tree\_Old'];

$four\_Old = $\_POST['four\_Old'];

$tbName = $\_POST['tblName'];

$err = "";

$err = ValidateFlats($one\_Old, $two\_Old, $tree\_Old, $four\_Old, $tbName);

if($err != "") echo($err);

else{

if($tbName == 'Классы'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="UPDATE `Классы` SET `№ класса` = '$one', `Табельный № учителя` = '$two', `Специализация класса` = '$tree', `Классная комната` = '$four'

WHERE `№ класса` = '$one\_Old' AND `Табельный № учителя` = '$two\_Old' AND `Специализация класса` = '$tree\_Old' AND `Классная комната` = '$four\_Old';";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно обновить запись из таблицы \"Классы\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись с ключевым полем ".$one." успешно обновлена</p>\n");

}

else if($tbName == 'Работники школы'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="UPDATE `Работники школы` SET `Табельный номер` = '$one', `ФИО учителя` = '$two', `Должность/профессия` = '$tree', `Пол` = '$four'

WHERE `Табельный номер` = '$one\_Old' AND `ФИО учителя` = '$two\_Old' AND `Должность/профессия` = '$tree\_Old' AND `Пол` = '$four\_Old';";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно обновить запись из таблицы \"Работники школы\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись с ключевым полем ".$one." успешно обновлена</p>\n");

}

else if($tbName == 'Школьники'|| $tbName == 'Школьники1'){

//echo ("<p>сейчас должен вставить</p>\n");

$query="UPDATE `$tbName` SET `№ класса` = '$one', `Порядковый № в журнале` = '$two', `ФИО школьника` = '$tree', `Пол` = '$four'

WHERE `№ класса` = '$one\_Old' AND `Порядковый № в журнале` = '$two\_Old' AND `ФИО школьника` = '$tree\_Old' AND `Пол` = '$four\_Old';";

$result=$conn->query($query);

if (!$result)

echo ("<p>Невозможно обновить запись из таблицы \"".$tbName."\":<br>\n".$conn->error."</p>\n");

else

echo ("<p>Запись с ключевыми полями ".$one." и ".$two." успешно обновлена</p>\n");

}

}

}

$conn->close();

}

}

?>

В самом начале создается функция **«ValidateFlats()»** с тремя параметрами, которые соответствуют полям формы, а так же с четвертым параметром, который соответствует названию выбранной таблицы. Точно также, как и на стороне клиента через JS, здесь создается конструкция if-else для выбора шаблона в соответствии с таблицей, а так же создается переменная, которая будет записывать в себя ошибки, если они будут выявлены в ходе проверки. Условия if аналогичны условиям из JS, используются другие функции, встроенные в PHP, но выполняющие те же самые действия, что и функции из JS.

Скрипт начинается с попытки подсоединения к БД в PhpMyAdmin. Если все в порядке, и подключение состоялось, будет выведено сообщение об успехе и продолжится работа скрипта, если же возникли какие-то проблемы, действие скрипта прекращается и выводится сообщение об ошибке.

Далее, в случае успеха с подключением к БД, из скрытых полей формы ввода данных **«Ok»** и **«tblName»** с помощью функции **$\_POST[‘id’]** вынимаются значения с выбранным действием и названием таблицы соответственно, и записываются в переменные **$Ok** и **$tbName** для дальнейшей работы с ними. Так же из полей формы ввода данных вынимаются значения, введенные пользователем, и записываются в переменные **$one, $two, $tree и $four**. Далее создается переменная **$err**, в которую записывается результат функции **«ValidateFlats()» от переменных $one, $two, $tree, $four и $tbName.** В случае, если введенные пользователем данные окажутся некорректными или не были введены вовсе, пользователю будет выведен текст ошибки. Если же переменная **$err** окажется пустой, открывается конструкция if-elseif, которая проверяет значение переменной **$Ok** и определяет, какое действие с данными нужно совершить: удаление, изменение или добавление. Внутри каждго блока с проверкой выбранного действия так же находится конструкция if-elseif, которая с помощью переменной **$tbName**  проверяет к какой таблице должно производиться действие. В случае успешного выполнения действия по добавлению, удалению или изменению данных в таблице или же в случае неудачи, на экран пользователю выведется сообщение об итоге операции. Стоить упомянуть, что блок обновления данных отличается от блоков удаления и добавления тем, что в нем дополнительно производится извлечение «старых» значений таблицы из скрытых полей формы ввода данных **«one\_Old», «two\_Old», «tree\_Old» и «four\_Old»**, которые записываются в переменные **$one\_Old, $two\_Old, $tree\_Old и $four\_Old** соответственно.Это нужно для того, чтобы корректно указать, какую именно запись скрипт должен изменить.[27-28].

Если кнопкой, нажатой пользователем, будет «Добавить запись», тогда в переменную **$Ok** будет записано значение «insert», и произойдет выполнение блока кода для этой операции.

На рисунке 2.6.1 продемонстрирована попытка добавления записи с табельным номером учителя, которого нет в родительской таблице «Работники школы».

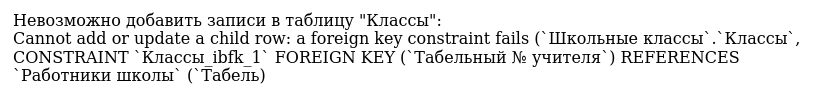


Рисунок 2.6.1 – Попытка добавить запись с несуществующим табельным № учителя

На рисунке 2.6.2 представлена попытка добавления дубликата [29].

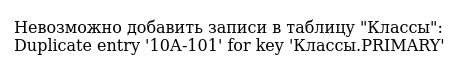


Рисунок 2.6.2 – Попытка добавления дубликата в таблицу «Классы»

Если была нажата кнопка «Удалить запись», происходит выполнение другого блока кода, соответствующего значению «delete».

На рисунке 2.6.3 продемонстрирована попытка удаления несуществующей записи.



Рисунок 2.6.3 – Попытка удаления несуществующей записи из таблицы «Классы»

Если код ошибки будет 1451, значит, у выбранной записи есть какие-то связи с дочерней таблицей и пользователю будет выведено сообщение об ошибке. Ниже на рисунке 2.6.4 продемонстрирована обработка исключения для удаления [29].

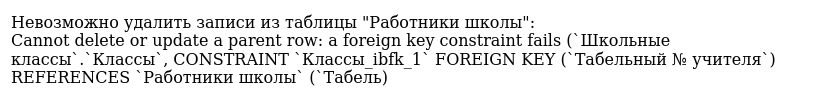


Рисунок 2.6.4 – Демонстрация обработки исключения для «delete»

Если пользователь нажал на кнопку «Обновить запись», происходит выполнение блока кода с «update». В этом блоке понадобятся так называемые «старые» значения, которые пользователь намеревается обновить. Поэтому в самом начале выполнения идет проверка через функцию **ValidateFlats()** на то, что заменяемые данные вообще есть и отличны от значения null.

Ниже на рисунке 2.6.5 продемонстрирована обработка исключения [29].

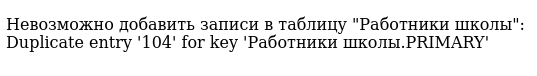


Рисунок 2.6.5 – Попытка добавления дубликата при обновлении записи

Данный php-скрипт вызывается на главной странице **«index.php»** в блоке **<section></section>** при помощи тега добавления php-скрипта **<?php?>** и вызова **«IDU.php»** внутри него с помощью php-метода **include\_once**. На рисунке 2.6.6 продемонстрирована печать таблицы «Работники школы» до изменений [30].

Для демонстрации работы кнопок будет произведено добавление нескольких записей, их обновление и последующее удаление. Рисунок 2.6.7 демонстрирует успешное добавление новых записей в таблицу «Работники школы».

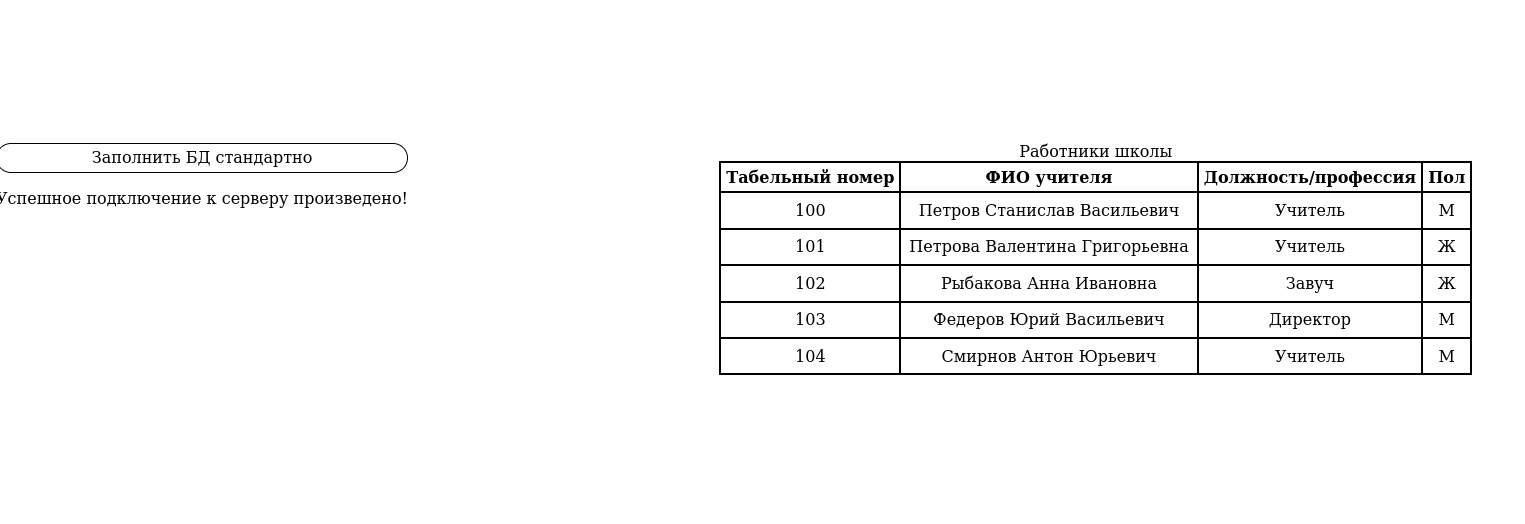


Рисунок 2.6.6 – Вывод таблицы «Работники школы»

В результате операции «Добавить запись» были созданы две записи. На рисунке 2.6.8 продемонстрировано успешное обновление нескольких записей.

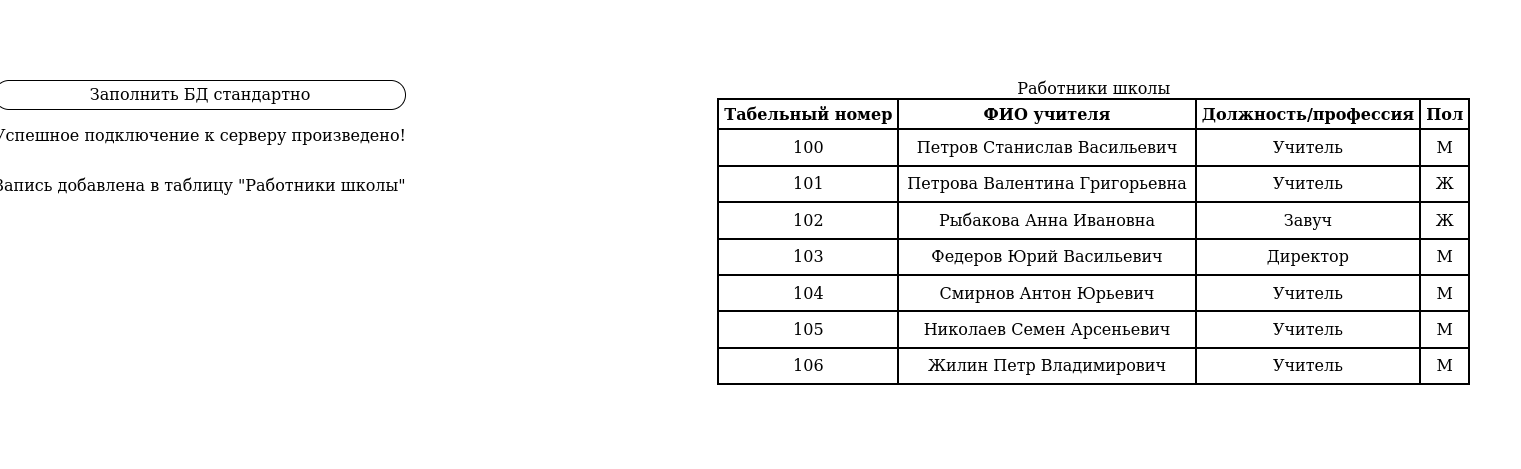


Рисунок 2.6.7 – Пример успешного добавления записей в таблицу «Работники школы»

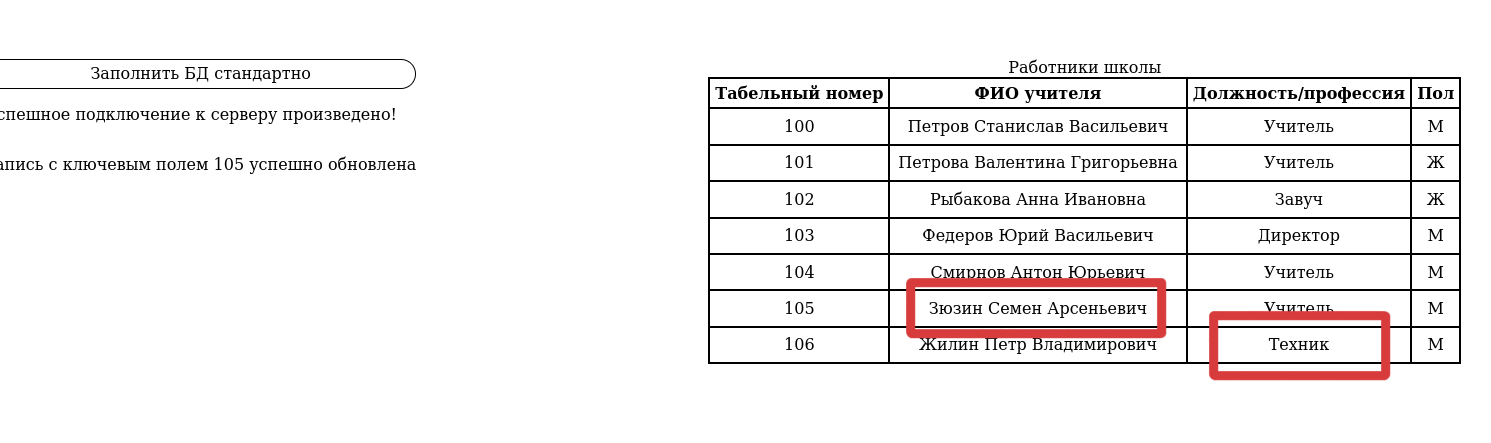


Рисунок 2.6.8 – Пример успешного обновления записей в таблице «Работники школы»

В результате операции «Обновить запись» в одной строке была изменена фамилия учителя, а во второй был изменена должность. На рисунке 2.6.9 изображена таблица после успешного удаления выбранной записи.

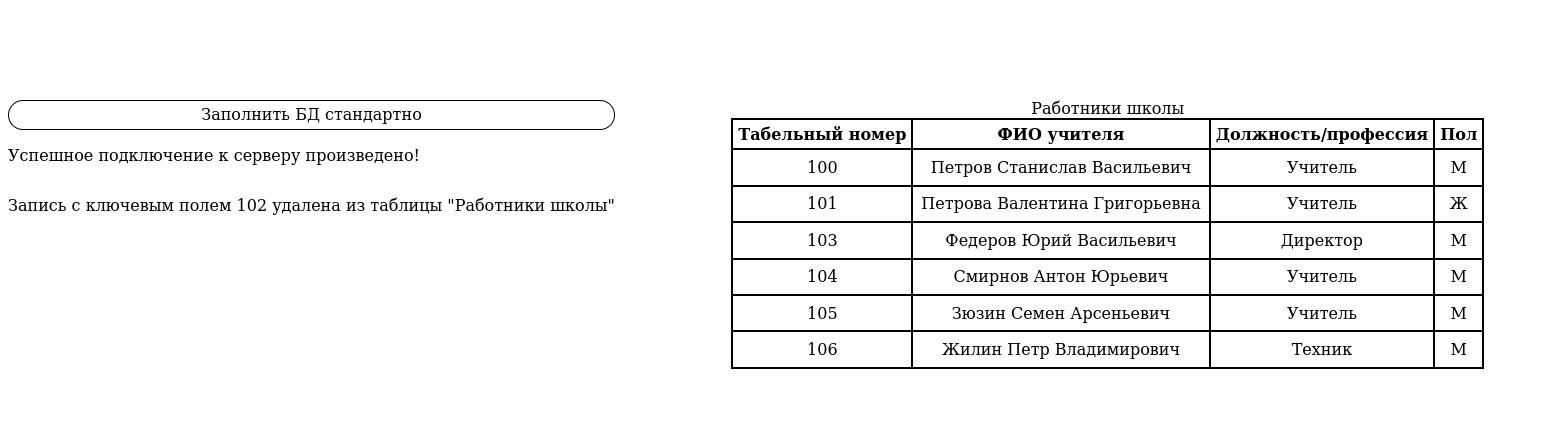


Рисунок 2.6.9 – Пример успешного удаления записи из таблицы «Работники школы»

В результате операции «Удалить запись», была удалена запись об учителе с табельным номером 102.

По окончании выполнения блока кода для выбранной операции и вывода обновленной таблицы соединение с сервером MySQL закрывается при помощи вызова встроенной функции «mysqli\_close», в аргументы которой передается изначальный адрес соединения.

## **2.7 Создание сводной таблицы**

При нажатии на кнопку «Отчет» на главной странице приложения пользователю будет выведена форма с двумя кнопками для выбора вида отчета. Ниже представлен листинг программного кода на создание этой формы (весь код скрипта расположен в папке forms в файле form.html). В области, в которой раньше по нажатию синхронно с формой появлялась таблица, теперь ничего не будет.

<div>

<button onclick="getDate('selectOtchSHK.php','tables')">Показать отчет для Школьники</button>

<button onclick="getDate('selectOtchSHK1.php','tables')">Показать отчет для Школьники1</button>

</div>

Рисунок 2.7.1 демонстрирует созданную форму.

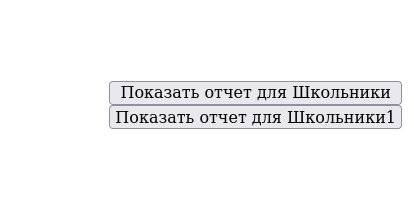


Рисунок 2.7.1 – Форма для вывода отчета

Нажатие кнопки «Показать отчет для Школьники» на форме выведет пользователю таблицу, в которой будет сводная таблица, связывающая таблицу Школьники со всеми остальными. По аналогии сделана кнопка «Показать отчет для Школьники1» На рисунках 2.7.2 и 2.7.3 представлен результат нажатия на кнопки.

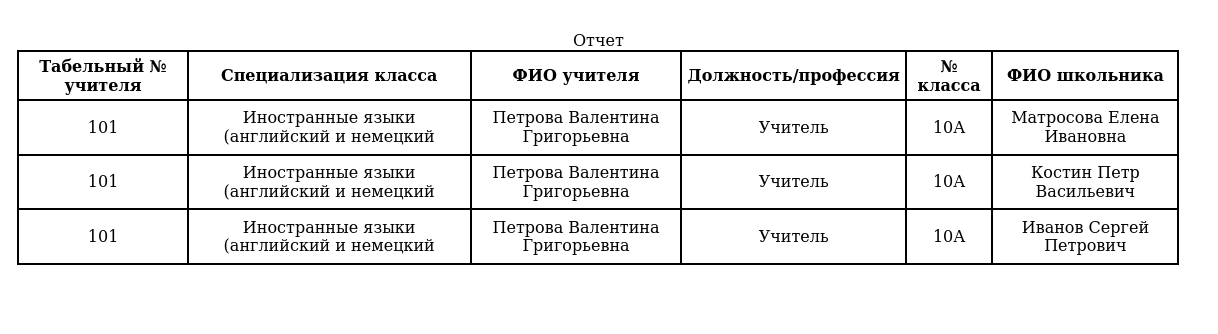


Рисунок 2.7.2 – Сводная таблица Школьники



Рисунок 2.7.3 – Сводная таблица Школьники 1

# **ЗАКЛЮЧЕНИЕ**

В современном мире такая вещь, как интернет, является незаменимым инструментом. Без него было практически невозможно общаться с людьми из других стран и с других континентов. Интернет может помочь некоторым людям отвлечься от внешнего мира. При помощи этой сети постоянно происходят операции различной степени сложности, от отправки видеороликов своим друзьям до финансовой транзакции.

Человеку не нужно ждать газету, можно просто зайти в интернет и узнать интересующие новости. Именно из интернета, по большей части, люди получают какие-то новые знания. Если человек собирается отправиться куда-то в путешествие, интернет поможет ему проложить путь к месту назначения.

Современное поколение очень зависимо от интернета, и жизнь без него сможет представить лишь небольшая группа людей. Жизнь многих без интернета станет настолько скучной и неинтересной, что они окажутся не в «своей тарелке».

В результате выполнения курсовой работы были освоены следующие области веб-разработки:

* основы языка разметки страницы HTML;
* каскадная таблица стилей CSS;
* основы языка программирования JavaScript;
* основы языка программирования PHP;
* работа со средой MySQL;
* работа по технологии AJAX.

В процессе написания курсовой работы были освоены следующие компетенции:

* ПК-4 Проектирование и дизайн ИС;
* ПК-5 Организационное и технологическое обеспечение кодирования на языках программирования и модульного тестирования ИС.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Значение интернета в 21-ом веке. // <http://supreme2.ru> URL: <http://supreme2.ru/7937-znachenie-interneta-v-xxi-veke/> (дата обращения сайт 23.04.2022). Режим доступа: свободный.
2. Интернет – трамплин в развитии человечества. // <https://sci-news.ru> URL: <https://sci-news.ru/2019/internet-i-ego-rol-v-razvitii-chelovechestva/> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
3. Как интернет изменил жизнь человека. // <https://mgts.ru> URL: <https://mgts.ru/articles/internet/Kak_internet_izmenil_obraz_zhizni_sovremennyh_lyudej/> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
4. Основы HTML. // <https://html5book.ru> URL: <https://html5book.ru/osnovy-html/> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
5. Обучение HTML. // <https://www.w3schools.com> URL: <https://www.w3schools.com/html/default.asp> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
6. Каскадная таблица стилей CSS. // <https://developer.mozilla.org> URL: <https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
7. Современный учебник JavaScript. // <https://learn.javascript.ru> URL: <https://learn.javascript.ru/> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
8. Обучение JavaScript. // <https://learn.javascript.ru> URL: <https://learn.javascript.ru> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
9. Достоинства и недостатки JavaScript. // <https://vc.ru> URL: <https://vc.ru/hr/145461-yazyk-programmirovaniya-javascript-osobennosti-i-preimushchestva?ysclid=l2fzxx0sq5> (дата обращения на сайт 23.04.2022). Режим доступа: свободный.
10. Что такое PHP. // <https://www.php.net> URL: <https://www.php.net/manual/ru/intro-whatis.php> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
11. Возможности PHP. // <https://www.php.net> URL: <https://www.php.net/manual/ru/intro-whatcando.php> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
12. Особенности языка программирования PHP. // <https://htmlacademy.ru> URL: <https://htmlacademy.ru/tutorial/php/basics?ysclid=l2g030tc7b> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
13. Введение в технологию AJAX. // <https://javascript.ru> URL: <https://javascript.ru/ajax/intro> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
14. Преимущества технологии AJAX. // <https://wiki.rookee.ru> URL: <https://wiki.rookee.ru/ajax/> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
15. Этапы создания запросов по технологии AJAX. // <https://itchief.ru> URL: <https://itchief.ru/javascript/ajax-introduction> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
16. Что такое XAMPP и что он в себя включает. // <https://developernotes.ru> URL: <https://developernotes.ru/xampp/chto-eto-takoe-xampp-podrobnoe-obyasnenie> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
17. Веб-сервер Apache. // <https://thecode.media> URL: <https://thecode.media/apache/> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
18. Функция mysqli\_close. // <https://www.php.net> URL: <https://www.php.net/manual/ru/mysqli.close.php> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
19. Среда PhpMyAdmin. // <https://ipipe.ru> URL: <https://ipipe.ru/info/phpmyadmin> (дата обращения на сайт 24.04.2022). Режим доступа: свободный.
20. Отличие синхронных запросов от асинхронных. // <https://www.internet-technologies.ru> URL: <https://www.internet-technologies.ru/articles/kogda-ispolzovat-asinhronnyy-ili-sinhronnyy-ajax.html> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
21. Обработчик событий в JavaScript. // <https://www.internet-technologies.ru> URL: <https://www.internet-technologies.ru/articles/obrabotka-sobytiy.html> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
22. Поля типа hidden. // <http://htmlbook.ru> URL: <http://htmlbook.ru/samhtml5/formy/skrytoe-pole> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
23. Проверка формы на стороне клиента. // <https://msiter.ru> URL: <https://msiter.ru/tutorials/javascript/js_validation> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
24. Функция isNaN в JavaScript. // <https://javascript.ru> URL: <https://javascript.ru/isnan> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
25. Регулярные выражения RegEx. // <https://regex.sorokin.engineer> URL: <https://regex.sorokin.engineer/ru/latest/regular_expressions.html> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
26. Объекты FormData. // <https://learn.javascript.ru> URL: <https://learn.javascript.ru/formdata> (дата обращения на сайт 25.04.2022). Режим доступа: свободный.
27. Функция isset. // <https://www.php.net> URL: <https://www.php.net/manual/ru/function.isset.php> (дата обращения на сайт 26.04.2022). Режим доступа: свободный.
28. Ассоциативный массив $\_POST. // <https://www.php.net> URL: <https://www.php.net/manual/ru/reserved.variables.post.php> (дата обращения на сайт 26.04.2022). Режим доступа: свободный.
29. Исключения в PHP. // <https://www.php.net> URL: <https://www.php.net/manual/ru/language.exceptions.php> (дата обращения на сайт 26.04.2022). Режим доступа: свободный.
30. Функция mysqli\_fetch\_all. // <https://www.php.net> URL: <https://www.php.net/manual/ru/mysqli-result.fetch-all.php> (дата обращения на сайт 26.04.2022). Режим доступа: свободный.

# **ПРИЛОЖЕНИЕ А**

Программный код дизайна созданного веб-приложения (файл style.css).

/\* PAGE \*/

li {

list-style: none;

}

/\* HEADER ////////////////////////////////////////////////////////////////////////////\*/

header {

display: flex;

color: rgb(17, 13, 13);

align-items: center;

justify-content: space-between;

padding: 0 20px;

}

.logo {

height: 60px;

}

.nav {

display: flex;

gap: 40px;

}

.nav\_\_item {

text-decoration: none;

color: rgb(0, 0, 0);

border: 1px solid black;

border-radius: 25px;

padding: 5px;

background-color: rgb(255, 255, 255);

}

.nav\_\_item:hover {

background-color: rgba(0, 0, 0, 0.6);

color: aliceblue;

}

.adress {

text-decoration: none;

color: rgb(59, 55, 55);

}

.adress\_\_icon {

width: 20px;

}

.info {

padding-left: 110px;

;

}

.info\_list {

color: rgb(59, 55, 55);

}

/\* MAIN ////////////////////////////////////////////////////////////////////////////\*/

main {

display: flex;

/\* align-items: center; \*/

padding-left: 20px;

padding-right: 50px;

justify-content: space-between;

padding-bottom: 200px;

}

table,

td,

th {

border: 1px solid black;

float: center;

}

th {

padding: 5px;

}

section {

padding-top: 200px;

}

.vid {

max-width: 45%;

display: flex;

flex-direction: column;

}

.vb {

float: center;

}

video {

position: absolute;

width: 100%;

object-fit: cover;

}

.mask {

position: absolute;

width: 100%;

z-index: 2;

user-select: none;

mix-blend-mode: screen;

}

article {

float: center;

padding-right: 10px;

padding-left: 25px;

display: flex;

flex-direction: column;

}

/\* ASIDE ////////////////////////////////////////////////////////////////////////////\*/

aside {

float: right;

display: flex;

flex-direction: column;

gap: 10px;

padding-left: 10px;

padding: 20px;

padding-top: 200px;

}

form {

float: center;

display: flex;

flex-direction: column;

}

div {

float: center;

display: flex;

flex-direction: column;

}

/\* FOOTER ////////////////////////////////////////////////////////////////////////////\*/

footer {

background-color: rgb(17, 17, 17);

width: 100%;

text-align: center;

color: azure;

padding: 10px 0;

position: absolute;

left: 0;

bottom: 0;

}

# **ПРИЛОЖЕНИЕ Б**

Программный код для отображения таблицы «Классы» (файл selectKL.php).

<?php

require\_once 'login.php';

$conn = new mysqli($server, $user, $password, $dbname);

if ($conn->connect\_error)

echo "<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."</p>\n";

else {

$query="SELECT `№ класса`, `Табельный № учителя`, `Специализация класса`, `Классная комната` FROM `Классы`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно просмотреть записи из таблицы \"Классы\": $conn->error </p>\n";

else {

echo "<table id='tbl' name='Классы' cellspacing='0' cellpadding='8' border='1' align='center'>\n";

echo "\t<caption id='tblCap'>Классы</caption>\n";

echo "\t<tbody style='text-align:center'>\n";

echo "\t\t<tr><th>№ класса</th><th>Табельный № учителя</th><th>Специализация класса</th><th>Классная комната</th></tr>\n";

while ($row=$result->fetch\_array(MYSQLI\_NUM)) {

echo "\t\t<tr><td>$row[0]</td><td class='stL'>$row[1]</td><td class='stL'>$row[2]</td><td>$row[3]</td></tr>\n";

}

echo "\t</tbody>\n";

echo "</table>\n";

$result->close();

}

$conn->close();

}

?>

Программный код для отображения таблицы «Работники школы» (файл selectRB.php).

<?php

require\_once 'login.php';

$conn = new mysqli($server, $user, $password, $dbname);

if ($conn->connect\_error)

echo "<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."</p>\n";

else {

$query="SELECT `Табельный номер`, `ФИО учителя`, `Должность/профессия`, `Пол` FROM `Работники школы`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно просмотреть записи из таблицы \"Работники школы\": $conn->error </p>\n";

else {

echo "<table id='tbl' name='Работники школы' cellspacing='0' cellpadding='8' border='1' align='center'>\n";

echo "\t<caption id='tblCap'>Работники школы</caption>\n";

echo "\t<tbody style='text-align:center'>\n";

echo "\t\t<tr><th>Табельный номер</th><th>ФИО учителя</th><th>Должность/профессия</th><th>Пол</th></tr>\n";

while ($row=$result->fetch\_array(MYSQLI\_NUM)) {

echo "\t\t<tr><td>$row[0]</td><td class='stL'>$row[1]</td><td class='stL'>$row[2]</td><td>$row[3]</td></tr>\n";

}

echo "\t</tbody>\n";

echo "</table>\n";

$result->close();

}

$conn->close();

}

?>

Программный код для отображения таблицы «Школьники» (файл selectSHK.php).

<?php

require\_once 'login.php';

$conn = new mysqli($server, $user, $password, $dbname);

if ($conn->connect\_error)

echo "<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."</p>\n";

else {

$query="SELECT `№ класса`, `Порядковый № в журнале`, `ФИО школьника`, `Пол` FROM `Школьники`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно просмотреть записи из таблицы \"Школьники\": $conn->error </p>\n";

else {

echo "<table id='tbl' name='Школьники' cellspacing='0' cellpadding='8' border='1' align='center'>\n";

echo "\t<caption id='tblCap'>Школьники</caption>\n";

echo "\t<tbody style='text-align:center'>\n";

echo "\t\t<tr><th>№ класса</th><th>Порядковый № в журнале</th><th>ФИО школьника</th><th>Пол</th></tr>\n";

while ($row=$result->fetch\_array(MYSQLI\_NUM)) {

echo "\t\t<tr><td>$row[0]</td><td class='stL'>$row[1]</td><td class='stL'>$row[2]</td><td>$row[3]</td></tr>\n";

}

echo "\t</tbody>\n";

echo "</table>\n";

$result->close();

}

$conn->close();

}

?>

Программный код для отображения таблицы «Школьники1» (файл selectSHK1.php).

<?php

require\_once 'login.php';

$conn = new mysqli($server, $user, $password, $dbname);

if ($conn->connect\_error)

echo "<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."</p>\n";

else {

$query="SELECT `№ класса`, `Порядковый № в журнале`, `ФИО школьника`, `Пол` FROM `Школьники1`";

$result=$conn->query($query);

if (!$result)

echo "<p>Невозможно просмотреть записи из таблицы \"Школьники1\": $conn->error </p>\n";

else {

echo "<table id='tbl' name='Школьники1' cellspacing='0' cellpadding='8' border='1' align='center'>\n";

echo "\t<caption id='tblCap'>Школьники1</caption>\n";

echo "\t<tbody style='text-align:center'>\n";

echo "\t\t<tr><th>№ класса</th><th>Порядковый № в журнале</th><th>ФИО школьника</th><th>Пол</th></tr>\n";

while ($row=$result->fetch\_array(MYSQLI\_NUM)) {

echo "\t\t<tr><td>$row[0]</td><td class='stL'>$row[1]</td><td class='stL'>$row[2]</td><td>$row[3]</td></tr>\n";

}

echo "\t</tbody>\n";

echo "</table>\n";

$result->close();

}

$conn->close();

}

?>

Программный код для отображения сводной таблицы для «Школьники» (файл selectOtchSHK.php).

<?php

require\_once 'login.php';

$conn = new mysqli($server, $user, $password, $dbname);

if ($conn->connect\_error)

echo "<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."</p>\n";

else {

$query="SELECT

`Классы`.`Табельный № учителя`, `Классы`.`Специализация класса`,

`Работники школы`.`ФИО учителя`, `Работники школы`.`Должность/профессия`,

`Школьники`.`№ класса`, `Школьники`.`ФИО школьника`

FROM `Работники школы` INNER JOIN `Классы` ON `Работники школы`.`Табельный номер` = `Классы`.`Табельный № учителя`

INNER JOIN `Школьники` ON `Классы`.`№ класса` = `Школьники`.`№ класса`

ORDER BY `Классы`.`Табельный № учителя`;";

$result=$conn->query($query);

if (!$result)

echo "<p>Во время выборки записей произошла ошибка: $conn->error </p>\n";

else {

echo "<table id='rep' name='Отчет' cellspacing='0' cellpadding='8' border='1' align='center'>\n";

echo "\t<caption id='tblCap'>Отчет</caption>\n";

echo "\t<tbody style='text-align:center'>\n";

echo "\t\t<tr><th>Табельный № учителя</th><th>Специализация класса</th><th>ФИО учителя</th><th>Должность/профессия</th><th>№ класса</th><th>ФИО школьника</th></tr>\n";

while ($row=$result->fetch\_array(MYSQLI\_NUM)) {

echo "\t\t<tr><td>$row[0]</td><td class='stL'>$row[1]</td><td class='stL'>$row[2]</td><td>$row[3]</td><td>$row[4]</td><td>$row[5]</td></tr>\n";

}

echo "\t</tbody>\n";

echo "</table>\n";

$result->close();

}

$conn->close();

}

?>

Программный код для отображения сводной таблицы для «Школьники1» (файл selectOtchSHK1.php).

<?php

require\_once 'login.php';

$conn = new mysqli($server, $user, $password, $dbname);

if ($conn->connect\_error)

echo "<p>Невозможно подключиться к серверу или открыть БД:<br>\n".$conn->connect\_error."</p>\n";

else {

$query="SELECT

`Классы`.`Табельный № учителя`, `Классы`.`Специализация класса`,

`Работники школы`.`ФИО учителя`, `Работники школы`.`Должность/профессия`,

`Школьники1`.`№ класса`, `Школьники1`.`ФИО школьника`

FROM `Работники школы` INNER JOIN `Классы` ON `Работники школы`.`Табельный номер` = `Классы`.`Табельный № учителя`

INNER JOIN `Школьники1` ON `Классы`.`№ класса` = `Школьники1`.`№ класса`

ORDER BY `Классы`.`Табельный № учителя`;";

$result=$conn->query($query);

if (!$result)

echo "<p>Во время выборки записей произошла ошибка: $conn->error </p>\n";

else {

echo "<table id='rep' name='Отчет' cellspacing='0' cellpadding='8' border='1' align='center'>\n";

echo "\t<caption id='tblCap'>Отчет</caption>\n";

echo "\t<tbody style='text-align:center'>\n";

echo "\t\t<tr><th>Табельный № учителя</th><th>Специализация класса</th><th>ФИО учителя</th><th>Должность/профессия</th><th>№ класса</th><th>ФИО школьника</th></tr>\n";

while ($row=$result->fetch\_array(MYSQLI\_NUM)) {

echo "\t\t<tr><td>$row[0]</td><td class='stL'>$row[1]</td><td class='stL'>$row[2]</td><td>$row[3]</td><td>$row[4]</td><td>$row[5]</td></tr>\n";

}

echo "\t</tbody>\n";

echo "</table>\n";

$result->close();

}

$conn->close();

}

?>

# **ПРИЛОЖЕНИЕ В**

Программный код формы ввода для таблицы «Работники школы» (папка forms, файл formRB.txt).

<form method="post">

<input type="hidden" id="Ok" name="Ok">

<input type="hidden" id="tblName" name="tblName">

<input type="hidden" id="one\_Old" name="one\_Old">

<input type="hidden" id="two\_Old" name="two\_Old">

<input type="hidden" id="tree\_Old" name="tree\_Old">

<input type="hidden" id="four\_Old" name="four\_Old">

<input name="one" id="one" placeholder="Табельный номер" required>

<input name="two" id="two" placeholder="ФИО учителя" required>

<input name="tree" id="tree" placeholder="Должность/профессия" required>

<input name="four" id="four"placeholder="Пол" required>

<button onclick="f1('insert')">Добавить выбранную запись</button>

<button onclick="f1('update')">Изменить выбранную запись</button>

<button onclick="f1('delete')">Удалить выбранную запись</button>

</form>

Программный код формы ввода для таблиц «Школьники» и «Школьники1» (папка forms, файл formSHK.txt).

<form method="post">

<input type="hidden" id="Ok" name="Ok">

<input type="hidden" id="tblName" name="tblName">

<input type="hidden" id="one\_Old" name="one\_Old">

<input type="hidden" id="two\_Old" name="two\_Old">

<input type="hidden" id="tree\_Old" name="tree\_Old">

<input type="hidden" id="four\_Old" name="four\_Old">

<input name="one" id="one" placeholder="№ класса" required>

<input name="two" id="two" placeholder="Порядковый № в журнале" required>

<input name="tree" id="tree" placeholder="ФИО школьника" required>

<input name="four" id="four" placeholder="Пол" required>

<button onclick="f1('insert')">Добавить выбранную запись</button>

<button onclick="f1('update')">Изменить выбранную запись</button>

<button onclick="f1('delete')">Удалить выбранную запись</button>

</form>

1. Ключевые слова подчеркнуты [↑](#footnote-ref-2)